



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

10/582,204

06/08/2006

Xiaofeng Guo

42P23159

4003

45209

7590

02/16/2011

MISSION/BSTZ

BLAKELY SOKOLOFF TAYLOR & ZAFMAN LLP

1279 OAKMEAD PARKWAY

SUNNYVALE, CA 94085-4040

EXAMINER

WEI, ZHENG

ART UNIT

PAPER NUMBER

2192

MAIL DATE

DELIVERY MODE

02/16/2011

PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

<b>Office Action Summary</b>	<b>Application No.</b>	<b>Applicant(s)</b>	
	10/582,204	GUO ET AL.	
	<b>Examiner</b>	<b>Art Unit</b>	
	ZHENG WEI	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 15 November 2010.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1-4,6-9 and 11-19 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-4,6-9 and 11-19 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- |  |   |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892)                     | 4) <input type="checkbox"/> Interview Summary (PTO-413)           |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____                                      |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08)          | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____  | 6) <input type="checkbox"/> Other: _____                          |

## **DETAILED ACTION**

### ***Remarks***

1. This office action is in response to the amendment filed on 11/15/2010.
2. Claims 5 and 10 have been canceled.
3. Claims 1-4, 6-9, 11-14 and 16-19 have been amended.
4. 35 U.S.C. 112 second paragraph rejection to claims 2, 5, 6, 10 and 14 is withdrawn in view of the Applicant's amendment.
5. 35 U.S.C. § 101 rejection to claims 9-12 and 16-19 is withdrawn in view of Applicants amendment.
6. Objection to claims 16-19 is withdrawn in view of Applicants' amendment.
7. Claims 1-4, 6-9 and 11-19 remain pending and have been examined.

### ***Response to Arguments***

8. Applicant's arguments filed on 11/15/2010, in particular on pages 9-14, have been fully considered but they are not persuasive. For example:
  - At page 11, last paragraph – page 12, first paragraph, Applicants submit "Li expressly and unambiguously states that the control flow graphs are used to form CFG loops are not based on 'inserting a plurality of dependency relationships based on the dependency graph between the plurality of blocks' as claimed by Applicants...The DFG disclosed by Li is not implemented using a plurality of blocks, nor is CFG an equivalent to dependency graphs as Applicants recite". However it should be noted that claim language does not

specify or define the CFG or dependency graph that has to be based on the 'inserting a plurality of dependency relationships based on the dependency graph between the plurality of blocks'. The limitation "inserting a plurality of dependency relationships" as recited can be reasonable interpreted as adjusting or rearranging the constructed CFG as disclosed by Li. Examiner's position is that Li discloses a plurality of basic blocks (CFG nodes) to construct a control flow graph (CFG) and basic blocks are organized logically (see for example, paragraph [0045], "As described herein, CFG nodes are basic blocks (sections of code always executed in order) and the edges represent possible flow of control between basic blocks").

- At page 12, first paragraph, Applicants submit that "Lauterbach does not disclose and is not offered to disclose the 'inserting a plurality of dependency relationships based on the dependency graph between the plurality of blocks' limitation as claimed by Applicants". However, Examiner's position is that Lauterbach discloses a method to build dynamic dependency graph (Fig.2, step 50) and insert dependencies to modify the dependency graph in order to identify performance results (see for example, col.1, lines 10-14). Li discloses a method to construct a control flow graph. Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to modify Li with Lauterbach's method by using the dependency graph to rearrange the computer program/blocks in Li. One would have been motivated to do so to use different methods (CFG or dependency graph

implementations) to identify different performance results as suggested by Lauterbach (col.1, lines 10-14) and improve performance as suggested by Li (paragraph [0022]).

- At page 14, first paragraph, Applicants submit that Lauterbach's dependency graph generator is not the same as 'constructing the dependency graph based on the organization of the plurality of blocks in the computer program' as Applicants recite". However, Examiner's position is that Lauterbach's dependency graph generation method can be used to modify Li's disclosure to construct a dependency graph based on Li's plurality of basic blocks.
- At page 14, second paragraph Applicants submit that "both the 'organizing the computer program logically into a plurality of blocks' and also that the 'constructing the dependency graph based on the organization of the plurality of blocks in the computer program' is specified to the multi-threaded programming instructions". However, it should be noted claim language does not recite such limitation about "the multi-threaded programming instructions".

### ***Claim Rejections - 35 USC § 103***

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

10. Claims 1-4, 6-9 and 11-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Li (Li et al., US 2005/0108695A1) in view of Lauterbach (Gary R. Lauterbach, US 5,712,791)

Claim 1:

Li discloses a computer implemented method for rearranging a computer program comprising:

- organizing the computer program into a plurality of blocks (basic blocks) (see for example, pg [0039], "Fig.5 is a flowchart illustrating a method 500 for thread-partitioning a sequential application program"; also see Fig.2A-2C and related text; also see paragraph [0045], "CFG nodes are basic blocks...and the edges represent possible flow of control between basic blocks" and related text);
- constructing a control flow graph based on the organization of the plurality of blocks in the computer program (see for example, Fig.5, step 502, "Build a control flow graph (CFG) for a loop body of sequential application program to form a CFG loop" and related text; also see paragraph [0045], "CFG nodes are basic blocks...and the edges represent possible flow of control between basic blocks" and related text);
- determining a critical section included in the CFG (see for example, paragraph [0042], "an identified critical section of the sequential application program is selected" and related text);

- detecting a portion of the computer program that could be executed outside of the critical section (see for example, paragraph [0046], “...code motion moves irrelevant code out of identified critical section...motion candidate instructions are identified using dataflow analysis” and related text)
- inserting a plurality of dependency relationships between the plurality of blocks to cause execution of the detected portion of the plurality of blocks in the computer program outside of the critical section (see for example, Fig.5, step 510, “update nodes of the CFG loop to enclose identified critical sections of the sequential application program within pairs of boundary instructions” and related text) paragraph [0046], “...code motion is a technique for inter-block and intra-block instruction reordering (hoisting/sinking)...code motion moves irrelevant code out of identified critical sections in order to minimize the amount of instruction/operations contained therein”);
- rearranging the detected portion of the plurality of blocks to outside the critical section that were inside the critical section based on the inserted plurality of dependency relationships (see for example, Fig.5, step 520, “Modify nodes of the CFG Loop to Reduce an amount of instructions between corresponding pairs of bonding instructions to form a modified CFG loop” and related text)

Li discloses constructing a control flow graph, but discloses not explicitly discloses constructing the dependency graph based on the organization of the computer program. However, Lauterbach in the same analogous art discloses a method to generate dependency graph based on instructions (see for example,

Col.3, lines 26-27, "...the dependency graph generator generates a dependency graph for a set of program instruction"; Fig.2, step 50, "Build Dependency Graph for Trace of instructions" and related text; also see Fig.3 illustrates a dynamic dependency graph associated with a set of program instructions) and inserting dependency to modify the dependency graph to identify/improve different performance results (see for example, col.1, lines 10-14) . Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to construct the dependency graph using Lauterbach's method in Li's invention to rearrange the computer program. One would have been motivated to do so to use different methods (CFG or dependency graph) to rearrange program and identify related performance results as suggested by Lauterbach (col.1, lines 10-14) and improve performance as suggested by Li (paragraph [0022]).

Claim 2:

Li discloses the method of claim 1 wherein each of the plurality of blocks includes computer program instructions (see for example, paragraph [0045], "As described herein, CFG nodes are basic blocks (sections of code always executed in order) and the edges represent possible flow of control between basic blocks")

Claim 3:



Li discloses the method of claim 1 further comprises organizing the plurality of blocks in the computer program based on a node and a super block, wherein the node includes a plurality of blocks and the super block includes a plurality of nodes (see for example, Fig.2A-C, Fig.3A-B).

Claim 4:

Li discloses the method of claim 1, wherein the critical section included in the dependency graph accesses shared resources (loop carried variables) (see for example, paragraph [0034], “each loop carried variable is assigned within a unique critical section to synchronize access to the loop carried variables in order to form program-thread...”).

Claim 6:

Li discloses the method of claim 1 further comprising adding a termination point to the critical section if a portion of the critical section is outside of the dependency graph (see for example, Fig.3B, item 328 and related text).

Claim 7:

Li discloses the method of claim 1, but does not explicitly disclose inserting additional dependency relationship based on a direct dependency, an indirect dependency, or a shortest life-time dependency. However, Lauterbach in the

same analogous art discloses insert artificial dependencies into the generated dependency graph (see for example, Fig.2, step 56 “Insert Artificial dependencies into dependency graph”). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to use Lauterbach’s method to insert additional dependency relationship to add additional limitations as suggested Lauterbach (see for example, col.2, lines 11-13)

Claim 8:

Li discloses the method of claim 1 further comprises comprising scheduling to execute the plurality of blocks in the computer program based on the dependency graph, after rearranging the detected portion of the plurality of blocks to outside the critical section (see for example, Fig.16, step 594, “Concurrently execute the plurality of application threads within a respective thread of a multi-threaded architecture” and related text).

Claims 9 and 11-12:

Claims 9 and 11-12 are system version for performing the claimed method as in claims 1-4 and 6-8 addressed above, wherein all claimed limitation functions have been addressed and/or set forth above and certainly a computer system would need to run and/or practice such function steps disclosed by reference above. Thus, they also would have been obvious.

Claims 13-15:

Claims 13-15 are another system version for performing the claimed method as in claims 1-4 and 6-8 addressed above, wherein all claimed limitation functions have been addressed and/or set forth above and certainly a computer system would need to run and/or practice such function steps disclosed by reference above. Thus, they also would have been obvious.

Claims 16-19:

Claims 16-19 are computer program products version of the claimed method, wherein all claimed limitation functions have been addressed in claims 1-4 and 6-8 above respectively. It is well known in the computer art that such method steps can be implemented as computer program and can be practiced and /or stored on a machine-accessible medium. Thus, they also would have been obvious in view of reference teachings above.

***Conclusion***

11. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.
12. Applicant's arguments with respect to claims rejection have been fully considered but they are not persuasive. Applicant's amendment necessitated additional clarification and/or the new ground(s) of rejection presented in this Office action.

Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

13. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Zheng Wei whose telephone number is (571) 270-1059 and Fax number is (571) 270-2059. The examiner can normally be reached on Monday-Thursday 8:00-15:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature of relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571- 272-1000.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Z. W./  
Examiner, Art Unit 2192

/Tuan Q. Dam/  
Supervisory Patent Examiner, Art Unit 2192